Claims

What is claimed is:

1. A speech label accelerator (SLA) comprising:

an indirect memory adapted to store a fixed plurality of indexes corresponding to a fixed plurality of atom functions;

an atom value memory coupled to the indirect memory, the atom value memory adapted to store a fixed plurality of atom values corresponding to a fixed plurality of atom functions, wherein each of the indexes selects one of the atom values in the atom value memory, wherein each of the atom values is determined for a particular input vector and a particular atom function, and wherein the atom functions are selected to represent a plurality of kernel functions thereby providing an approximation to the plurality of kernel functions; and

adder circuitry coupled to the atom value memory, the adder circuitry adapted to add atom values selected by indexes of the indirect memory.

2. The SLA of claim 1, wherein each of the atom functions has domain R^d for some plurality of dimensions, d.

3. The SLA of claim 1, wherein each of the atom functions has domain R for a single dimension.

4. The SLA of claim 1, wherein the adder circuitry comprises a pipelined tree of adders.

5. The SLA of claim 4, wherein the pipelined tree of adders comprises a plurality of stages, each of the stages comprising a plurality of adders and at least one register.

6. The SLA of claim 5, wherein a number of dimensions is denoted by d, wherein n is a stage number, wherein each stage comprises $\lfloor d/(2^n) \rfloor$ sums in parallel, and wherein there are $\lceil \log_2 d \rceil$ stages.

7. The SLA of claim 6, further comprising an accumulator coupled to the adder circuitry, wherein the adder circuitry further comprises a final stage when d is an integral power of two.

8. The SLA of claim 1, further comprising an accumulator coupled to the adder circuitry, the accumulator adapted to accumulate at least one result of additions between atom values.

9. The SLA of claim 1, wherein the adder circuitry comprises a pipelined adder chain.

10. The SLA of claim 9, wherein the pipelined adder chain comprises a number of single dimension adders, wherein the number of single dimension adders is the same as the number of dimensions.

11. The SLA of claim 10, wherein each single dimension adder comprises an adder adding a previous dimension adder output to a selected atom value, wherein the

selected atom values are input to each of the single dimension adders in parallel.

12. The SLA of claim 1, further comprising a load/accumulate multiplexer (mux) and an accumulator having an input and output, the load/accumulate mux having two inputs and an output, wherein the adder circuitry comprises a first and second adder, each having two inputs and an output, the inputs of the first adder coupled to the atom value memory, a first input of the second adder coupled to the output of the first adder, a second input of the second adder coupled to the output of the load/accumulate mux, an input of the load/accumulate mux coupled to the output of the accumulator, and the output of the second adder coupled to the input of the accumulator.

13. The SLA of claim 12, wherein the accumulator comprises a demultiplexer (demux), a mux, and a plurality of registers, the demux having an input and a plurality of outputs, the mux having a plurality of inputs and an output, the input of the demux coupled to the input of the accumulator, each of the registers coupled to one output of the demux and to one input of the mux, and the output of the mux coupled to the output of the accumulator.

14. The SLA of claim 8, further comprising a load/accumulate multiplexer (mux) having a first input coupled to the accumulator, an output coupled to an input of the adder circuitry, and a second input coupled to a zero value.

15. The SLA of claim 14, further comprising a control unit coupled to the indirect memory, atom value memory, adder circuitry, and accumulator.

16. The SLA of claim 1, wherein the atom and kernel functions are logarithms of other functions.

17. The SLA of claim 1, wherein the kernel functions are completely separable.

18. The SLA of claim 1, wherein the kernel functions are partially separable.

19. The SLA of claim 1, wherein the atom and kernel functions are Gaussian functions.

20. The SLA of claim 1, wherein the atom and kernel functions are non-Gaussian functions.

21. The SLA of claim 1, wherein the atom functions are mixtures of Gaussians and the kernel functions are compound Gaussian functions.

22. A system comprising:

a processor;

a memory coupled to the processor; and

a speech label accelerator (SLA) coupled to the processor and the memory, the SLA comprising:

an indirect memory adapted to store a fixed plurality of indexes corresponding to a fixed plurality of atom functions;

an atom value memory coupled to the indirect memory, the atom value memory adapted to store a fixed plurality of atom values

corresponding to a fixed plurality of atom functions, wherein each of the indexes selects one of the atom values in the atom value memory, wherein each of the atom values is determined for a particular input vector and a particular atom function, and wherein the atom functions are selected to represent a plurality of kernel functions thereby providing an approximation to the plurality of kernel functions; and

adder circuitry coupled to the atom value memory, the adder circuitry adapted to add atom values selected by indexes of the indirect memory.

23. A method comprising the steps of:

determining, for a particular input vector, a plurality of atom values, wherein each of the atom values is determined from an atom function that represents a plurality of kernel functions thereby providing an approximation to the plurality of kernel functions;

loading a portion of the plurality of atom values into an atom value memory adapted to store a fixed number of atom values;

loading a portion of a plurality of indexes into an indirect memory adapted to store a fixed number of indexes, each of the loaded indexes adapted to select one of the atom values in the atom value memory, each of the loaded indexes corresponding to one of a fixed number of kernel functions;

selecting at least one index from the indirect memory;

retrieving at least one atom value corresponding to the at least one selected index from the atom value memory, one atom value retrieved per selected index; and

accumulating the at least one retrieved atom value.

24. The method of claim 23, further comprising the step of determining the plurality of indexes.

25. The method of claim 23, further comprising the step of performing the steps of selecting, retrieving, and accumulating until all indexes corresponding to a selected one of the fixed number of kernel functions have been selected, wherein all atom values corresponding to the selected kernel function are accumulated.

26. The method of claim 23, wherein the atom value memory comprises a size comprising a fixed number of atom functions by a fixed number of dimensions and wherein the indirect memory comprises a size comprising a fixed number of kernel functions by a fixed number of dimensions.

27. The method of claim 23, wherein each of the atom functions has domain R^d for some plurality of dimensions, d.

28. The method of claim 23, wherein each of the atom functions has domain R for a single dimension.

29. The method of claim 25, wherein the step of performing the steps of selecting, retrieving, and accumulating until all indexes corresponding to a selected one of the fixed number of kernel functions have been selected further comprises the steps of performing the steps of selecting and retrieving in parallel for all indexes for the selected kernel function, whereby all atom values corresponding to indexes for the selected kernel function are retrieved in parallel, and performing the step of accumulating after all the atom values have been retrieved.

30. The method of claim 29, wherein the step of accumulating is performed by a pipelined adder chain, and wherein the step of accumulating further comprises the steps of adding one of the retrieved atom values to another of the retrieved atom values and adding a result of all additions for all atom values into an accumulator.

31. The method of claim 29, wherein the step of accumulating is performed by a pipelined tree of adders having a plurality of stages, and wherein the step of accumulating further comprises the steps of performing additions during each stage to add results from a previous stage, wherein the first stage adds a number of the atom values, and adding a result of all additions for all stages into an accumulator.

32. The method of claim 25, wherein the step of performing the steps of selecting, retrieving, and accumulating until all indexes corresponding to a selected one of the fixed number of kernel functions have been selected further comprises the steps of performing the steps of selecting and retrieving in parallel for every two indexes for the selected kernel function, and performing the step of accumulating after the two atom values have been retrieved.

33. The method of claim 23, wherein the step of accumulating the retrieved atom value further comprises the steps of:

adding the retrieved atom value to a value from an accumulator to create a result;

updating the accumulator with the result; and

setting the accumulator to zero prior to the step of selecting at least one index from the atom value memory.

34. The method of claim 25, wherein the method further comprises the step of selecting another of the kernel functions, and wherein the step of performing the steps of selecting, retrieving, and accumulating further comprises the steps of:

performing the steps of selecting, retrieving, and accumulating until all indexes for the selected kernel function have been selected; and

performing the previous step and the step of selecting another of the kernel functions until all the kernel functions have been selected, wherein all atom values, corresponding to indexes from the fixed number of kernel functions, are accumulated.

35. The method of claim 25, wherein:

the step of determining, for a particular input vector, a plurality of atom values further comprises the steps of

determining, for a particular input vector, a plurality of atom values corresponding to a plurality of atom functions having a first number of dimensions, wherein the first number of dimensions is larger than a number of dimensions of the atom value memory; and

separating the atom values into blocks, each block comprising atom functions having the number of dimensions of the atom value memory;

the step of loading of portion of the plurality of atom values into an atom value memory comprises the step of loading a selected one of the blocks into atom value memory; and

the step of performing the steps of selecting, retrieving, and accumulating further comprises the steps of:

performing the steps of selecting, retrieving, and accumulating until all indexes corresponding to the selected kernel function and to the selected block have been selected;

loading another selected block of atom values into the atom value memory; and

performing the two previous steps until all blocks have been selected, wherein all atom values corresponding to indexes for the selected kernel function are accumulated.

36. The method of claim 25, wherein:

the step of determining, for a particular input vector, a plurality of atom values further comprises the steps of:

selecting one of a plurality of hierarchical levels;

determining, for a particular input vector, a plurality of atom values corresponding to a plurality of atom functions having a first number of dimensions, wherein the first number of dimensions is larger than a number of dimensions of the atom value memory;

separating the atom values into blocks, each block comprising atom functions having the fixed number of dimensions; and

assigning the blocks an order;

the step of loading a portion of the plurality of atom values into the atom value memory comprises the step of loading a selected one of the blocks into atom value memory; and

the step of performing the steps of selecting, retrieving, and accumulating further comprises the steps of:

performing the steps of selecting, retrieving, and accumulating until all indexes for the selected block of the selected kernel function have been selected;

selecting a next highest ordered block as the selected block and loading the selected block into the atom value memory;

performing the two previous steps until all blocks have been selected, wherein all atom values corresponding to indexes for the selected kernel function and for the selected level are accumulated, wherein the last block in the order resides in the atom memory;

selecting another of the hierarchical levels;

performing the steps of selecting, retrieving, and accumulating until all indexes for the selected block of the selected kernel function have been selected;

selecting a next lowest ordered block of atom values as the selected block loading the selected block into the atom value memory;

performing the three previous steps until all blocks have been selected, wherein all atom values corresponding to indexes for the selected kernel function and for the selected level are accumulated.

37. The method of claim 25, wherein:

the step of determining, for a particular input vector, a plurality of atom values further comprises the steps of

determining, for a particular input vector, a plurality of atom values corresponding to a plurality of atoms having a first number of dimensions, wherein the first number of dimensions is larger than a number of dimensions of the atom value memory; and

separating the atom values into blocks, each block comprising atom functions having the number of dimensions of the atom value memory;

the step of loading a portion of the plurality of atom values into the atom value memory comprises the step of loading a selected one of the blocks into atom value memory;

the step of performing the steps of selecting, retrieving, and accumulating further comprises the steps of:

performing the steps of selecting, retrieving, and accumulating until all indexes for one block of the selected kernel function have been selected;

loading another selected block of atom values into the atom value memory;

performing the two previous steps until all blocks have been selected, wherein all atom values corresponding to indexes for the selected kernel function are accumulated; and

selecting another of the kernel functions and performing the step of performing the steps of selecting, retrieving, and accumulating, the step of loading another selected block, and the step of performing the two previous steps until all of the kernel functions have been selected, wherein all atom values, corresponding to indexes from the kernel functions, for the all of the kernel functions are accumulated.

38. The method of claim 25, wherein:

the step of determining, for a particular input vector, a plurality of atom values further comprises the steps of

determining, for a particular input vector, a plurality of atom values corresponding to a first number of atoms, wherein the first number of atoms is larger than a number of dimensions of the atom value memory; and

separating the atoms into blocks, each block comprising the number of dimensions of the atom value memory, but wherein each of the atom values for one atom in each block is zero;

the step of loading a portion of the plurality of atom values into the atom value memory comprises the step of loading a selected one of the blocks into atom value memory;

the step of performing the steps of selecting, retrieving, and accumulating further comprises the steps of:

performing the steps of selecting, retrieving, and accumulating until all indexes for one block of the selected kernel function have been selected;

loading another selected block of atom values into the atom value memory; and

performing the two previous steps until all blocks have been selected, wherein all atom values corresponding to indexes for the selected kernel function are accumulated.

39. The method of claim 23, wherein each of the atom and kernel functions is a logarithm of another function.

40. The method of claim 23, wherein each of the kernel functions is completely separable.

41. The method of claim 23, wherein at least one of the kernel functions is partially separable.

42. The method of claim 23, wherein each of the atom and kernel functions is a Gaussian function.

43. The method of claim 23, wherein the atom and kernel functions are non-Gaussian functions.

44. The method of claim 23, wherein the atom functions are mixtures of Gaussians and the kernel functions are compound Gaussian functions.